

If you MUST frugal a filing system under software control, you will have to do the following things.

1. Implode the font
 - Otherwise the OS will get confused
2. Claim NMI
 - This makes existing holder of NMI shut up his hardware
3. Write Break type = CTRL
 - This is the most important piece of magic
 - Without it you cannot guarantee that the newly allocated workspace will be initialised correctly
4. Sideways Call #1 with Y=&0E
 - ROMs should reset their hardware at this point
5. Remove or Frugal ROMs as required
6. Sideways Call #1 with Y=&0E (again)
 - The first call was necessary because frugalged ROMs will not necessarily bother to reset their hardware and Removed ROMs certainly won't.
 - This second call is the one which actually allocates the required amount of public workspace.
7. Sideways Call #2 with Y=0 value returned from step 6
8. Write Y reg returned from s/w #2 to both OSHWM (OSbyte &B3, &B4)
9. Check keyboard ?
10. Sideways call #3 ?

[Steps 9 and 10 may be replaced with *DISC, *NET etc]

You may wish to restart the current language, or start BASIC, at this point, before returning control to the main application program.

If you cannot understand this specification, then you should not dare to dabble in these murky waters. Make your programs work even when PAGE is set at &2000.

Acorn User
UserRAM

Manual & Installation Guide

Version 1.0
25th June, 1985

STATIC WARNING

The UserRAM is sensitive to static electrical discharges. Carefully follow these instructions to avoid damaging your RAM.

DO NOT TOUCH YOUR UserRAM UNTIL YOU HAVE READ THE INSTALLATION INSTRUCTIONS.

Published by
SJ Research
108 Mill Road
Cambridge, CB1 2BD

Copyright © Philip, Spence-Jones and Associates Limited
All rights reserved

First Published June 1985

Written by Kim Spence-Jones (with a lot of help from his friends)

Set on RPI600 flowwriter using
Cubic PS daisywheel
and formatting software by John Cox

Acknowledgement

I would like to thank Dave Upton, who wrote the software and put up with endless changes to the specifications, the staff at Acorn User (especially Tony Quinn and Bruce Smith) for their help and encouragement, the staff at SJ Research for their many excellent suggestions and last but by no means least Maureen for her proof reading and her tolerance!

Packing List

Your UserRAM pack should contain:

- A UserRAM chip with red flying lead
- A single disc (Pack 1 and Pack 2 both contain only one disc)
- This manual
- A copy of "The Sideways ROM Book" (Pack 2 only)

Contents

What is sideways RAM?

Using UserRAM

Installing the UserRAM

- Reading the program disc.
- Installing the Hardware.
 - BEWARE
 - First things first
 - Find out what ROMs are in your machine
 - save copies of any ROMs you wish to use in RAM before removing them
 - The arrangement of Sideways ROM Sockets
 - Which socket should you use?
- STATIC PRECAUTIONS
- Getting In
 - Doing the Shuffle
- Plugging in Your Sideways RAM
 - Attaching the red lead
- Using More than One ROM
- Reassembly
- Testing the UserRAM
 - *RAMTEST
 - *DEMO
 - If the test fails ..

Pitfalls for the unwary!

- Externally powered peripherals
- Rapid cycling of the power switch
- Protected ROMs

Acorn User Support

The UserRAM Pack 1 Utility Programs - Detailed Specifications

- Copy40
- *RLIST
- *DEMO and R.DEMO
- *KILL [<ROM number>]
- *RLOAD [<filename> [<ROM number>]]
- MakeLd and *cromname
- *RSAVE [<filename> [<ROM number>]]
- *RCHECK [<filename> [<ROM number>]]
- *RTEST [<ROM number>]
- MakeRFS

The Acorn User Utility Programs

Appendix 1 : Summary of the Software supplied with UserRAM

Appendix 2 : The Frugalling Protocol

What is sideways RAM?

Small programs on the BBC micro are usually loaded into main memory. As the size and complication goes up it becomes less and less practical to do this, and it also becomes desirable to have the programs more permanently available. The original design of the BBC micro anticipated that this problem would be solved by fitting extra ROM chips inside the machine. Unfortunately only four suitable sockets are provided, two of which will be taken up by the Disc or Net Filing System and BASIC or other language. This leaves only two sockets free for other functions. Many people need more space than this for their extra ROMs.

There are two possible solutions to this problem. You can fit a ROM expansion board, which may allow up to 12 extra ROMs depending on the design. The disadvantage of this is that the expansion board is large and cumbersome, and all those extra chips may overload the power supply.

Sideways RAM is an alternative solution: instead of installing ROMs, you can fit one or more "pseudo ROMs" containing RAM which can be loaded when the BBC micro first needs a software package. The "ROM" will then be available just as if it was in a real ROM, except that it will be cleared when the BBC micro is turned off. This gives the added advantage that when you have finished with a particular piece of software you can overwrite it with something else. Instead of having only four or sixteen ROMs in your BBC Micro, you can effectively have as many as you like, loaded from disc or network whenever you need them.

To summarise, sideways RAM is a bank of memory which can be accessed for reading as if it was a sideways ROM, but which can also be written to. UserRAM is one type of sideways RAM.

Using UserRAM

In normal use, you will insert a disc with the appropriate ROM software on it, and type `*Name>`. The loader program that you will have already made (using `MakelD`) will then load the ROM from disc or net and start it as if it had been in ROM all the time. The ROM will then be available whenever required, until it is overwritten by something else.

You can also load a ROM for which you have not previously made a Loader program, using `*RLOAD <romname>`. This will not start the ROM, but you will be able to start it in the normal way, by typing `*BASIC`, `*LOGO`, `*IO` or whatever is appropriate.

To save a copy of a ROM, there is a program called `*RSAVE`. An image made in this way can immediately be loaded into UserRAM using `*RLOAD`, or you can use the BASIC program `MakelD` to make an automatic loader.

Some ROMs are protected to prevent you using them in RAM. These ROMs will corrupt themselves when they are started. The corruption can usually be detected using a BASIC program called `RCHECK`. We cannot give you any assistance in "converting" such ROMs for RAM use, and we strongly advise you not to try, as you are laying yourself open to prosecution under the Copyright Act.

Installing the UserRAM

Reading the program disc.

The program disc supplied with your UserRAM is written in 40-track format, on the top surface of the disc only. It can be read directly on any 40-track drive, or any 80-track drive with a 40/80-track switch. If you have an 80-track drive with no switch to read 40-track discs, you will have to copy the disc onto 80-track format before you can use it. A utility called `COPY40` is provided to allow you to do this, (and yes, you will be able to load it with your 80-track drive). Remember that it will only be necessary to use this if you cannot read 40 track discs directly.

If you need to make a copy, proceed as follows:

1. Format a new 80-track disc to receive the copy.
2. Insert the UserRAM disc.
3. Type `CHAIN "COPY40"`.
4. The program will prompt you for the source and destination drive numbers, which should be entered as a single digit between 0 and 3, followed by a `<return>`.
5. Unless you are copying from and to the same drive number the program will now complete the copy for you. If you are copying to the same drive, you will have to exchange discs in response to the prompts, in exactly the same way as when using `*COPY`.
5. The resulting disc should now have a standard set of programs on it, and can be used instead of the original in all instances.

See the full specification of `copy40` on page 14 for further details.

Installing the Hardware.

BEWARE

- a) If your micro is still under guarantee you may invalidate your guarantee rights by tampering with it.
- b) Electronic equipment is easy to damage if you are not careful. Read these instructions FULLY before starting to adjust your machine. In particular, the UserRAM chips are HIGHLY SENSITIVE TO THE STATIC ELECTRICAL CHARGES WHICH can build up on your body.

First things first

Before you reach for a screwdriver to open your micro, run the ROM menu program to find out what ROMs are currently in your micro, and which sockets they occupy. To do this, insert the UserRAM utilities disc and type:

`*RLIST <Return>`

You can make a hard copy of this if you have a printer available in the normal way, by typing `CTRL-B` before pressing `RETURN`. If not, then make a note of the contents of each ROM slot.

The arrangement of Sideways ROM Sockets

In the normal use of the computer there is no difference between the sockets, except:

1. Each socket has a different number, in the range 0 to 15
2. Calls to sideways ROMs are offered in order, from ROM 15 downwards. This is not normally important, unless you have two clashing command names, in which case the first ROM to receive the call will interpret the command, and it will not be easy to make the second ROM get the command.
3. On power up and CTRL-BREAK the language ROM with the highest number will automatically be started, and unless a key is held down on the keyboard whilst the BREAK key is released, the filing system with the highest numbered ROM will also be selected.

On an ordinary Model B BBC microcomputer there are four possible sockets for sideways ROMs. On the B+ there are five sockets, which can be configured to take 32 K x 8 ROMs configured as two 16 K x 8 ROMs. The ROM numbers of the sockets are given in the table below.

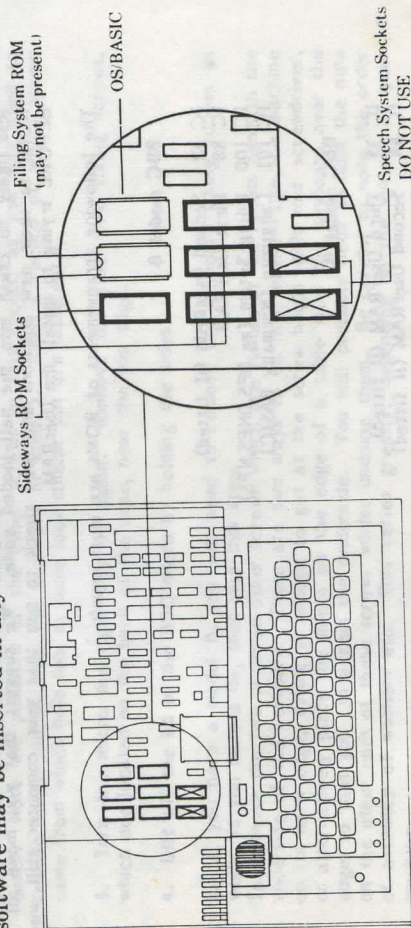
When you have sideways ROM boards fitted, the way the ROMs are numbered may be different, and it may be impossible to fit sideways RAM at all. Refer to the ROM board manufacturer's data for details.

BBC model B		BBC model B+	
Socket	ROM number	Socket	ROM number
IC 52	12	IC 35	2 &/or 3
IC 88	13	IC 44	4 &/or 5
IC 100	14	IC 57	6 &/or 7
IC 101	15	IC 62	8 &/or 9
		IC 68	10 &/or 11
		IC 71	OS & 14/15 (or OS & 0/1)

Table 1. The ROM numbers of Sideways ROMs

Inserting ROMs - BBC Microcomputer Model B+

This diagram shows a plan view of the BBC Microcomputer Model B+. The top of the computer casing has been removed to reveal the five sideways ROM sockets. ROM software may be inserted in any of these sockets.



Inserting ROMs - BBC Microcomputer Model B

This diagram shows a plan view of the BBC Microcomputer Model B. The top of the computer casing has been removed to reveal the four sideways ROM sockets. ROM software may be inserted in any of these sockets.

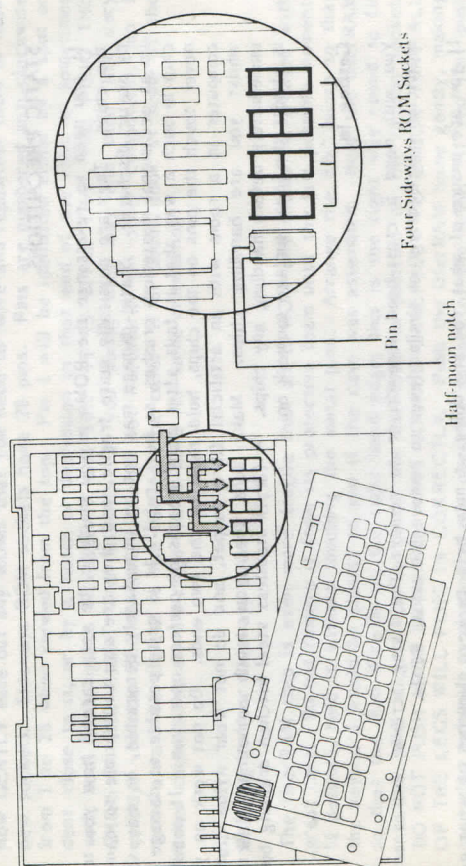


Figure 1. The Location of Sideways ROM Sockets.
Reproduced by kind permission of Acornsoft
© Acornsoft Limited, 1985

Which socket should you use?

There is no hard and fast rule as to which socket should be chosen for sideways RAM. We prefer to have at least one language (usually BASIC) and one filing system of higher priority than the RAM. This means that the RAM filing system or language will never be selected by default at CTRL-BREAK. The only real advantage of this is that if you press BREAK half-way through a UserRAM loading sequence you are less likely to crash into the half-loaded code. We suggest that you move all the ROMs to their new positions and then check to see that your computer still works, BEFORE trying to install the UserRAM.

The following arrangements of ROMs are most usual:

BBC model B

IC 52 Second UserRAM (if fitted)
IC 88 First UserRAM
IC 100 Filing System (eg DFS,DNFS,NFS)
IC 101 Language (Normally BASIC)

BBC model B+

IC 35 Third UserRAM (if fitted)
IC 44 Second UserRAM (if fitted)
IC 57 First UserRAM
IC 62 Second Language (if fitted)
IC 68 Filing System (eg DFS,DNFS,NFS)
IC 71 BASIC + OS in one ROM

When you have decided what order you wish to have your ROMs and RAM arranged in, make a sketch to remind you what you are trying to do.

STATIC PRECAUTIONS

If you need to re-arrange the ROMs in your machine, be very careful how you treat the chips. They are physically quite fragile, and they are also susceptible to damage by static discharge. Static damage may not be immediately detectable, or may show up as occasional mysterious crashes, so be careful. It is quite possible to damage the chip in such a way that it fails after many months of perfect operation. If possible, never touch the pins on the chips, hold them by the two ends. Do not work on your computer in a room with an artificial fibre carpet, and do not wear nylon clothes whilst you are handling chips. Make sure that you touch something "earthy" immediately before handling any chips. A (dry!) stainless steel kitchen draining board is almost the ideal surface to work on.

Getting In

You will need a cross-head screwdriver to remove the fixing screws, and a small electrician's screwdriver or similar lever to remove individual ROMs.

If you are unsure of what you are doing, it may help to keep sketches of what goes where, as you take it to pieces. Above all, try not to leave the job half finished, as it is much more difficult to remember what you are doing if you have to stop in the middle.

Proceed as follows:

1. Switch off your micro and unplug it from the mains. Disconnect all other peripherals from the micro, including the screen, printer etc.
2. Turn your micro upside down. Remove the two screws marked "fix" from the brown area under the keyboard (they are the two screws nearest to the edge of the case). Note the positions of the other screws in this recess, as you may need to remove these too to free the keyboard in step 5. Keep a record of which screw came from which hole too, as some look similar but are different.
3. Turn the micro the right way up again. Remove the other two fixing screws, which are located on the back of the case, near the top edge.
4. Lift off the lid of the BBC micro by holding the sides.
5. If you have a model B, you will need to remove the keyboard fixing screws as well. If you have a B+, then skip this step.
5a. Remove the two or three other screws which are visible in the recess beneath the underside of the keyboard. There are two ways to do this; either raise the machine on its back edge, tilting it enough to get at the screw heads with your screwdriver, or alternatively place the micro on the edge of a table with the keyboard over the edge, so that you can access the underside. You will probably need to hold the nuts on the other ends of these screws whilst undoing them. Be careful to note the order of assembly of washers etc. This varies greatly, depending on the vintage of your machine.
5b. It should now be possible to swivel the keyboard round to expose the ROM sockets, which are under the right hand end. The keyboard connector may come off when you do this, so note how it is plugged in so that you can reconnect it later if necessary. Be careful not to strain anything too much.

Doing the Shuffle

Now GENTLY ease out any ROMs that you need to move and reposition them in their new sockets. Sideways ROM sockets have 28 pins. Pins are numbered anti-clockwise from 1 to 28 when viewed from the top. Pin 1 will be identified either by a spot or dent close to it, or by a notch or depression in that end of the chip, or sometimes with both (see figures on pages 7 and 10). IT IS VITAL THAT YOU PLUG THE CHIPS IN THE RIGHT WAY ROUND. If you don't, they may end their life very quickly! In the BBC machine (though not necessarily on other computers), pin 1 should always be closest to the back of the case. If in doubt, look at some of the other chips, since all the chips should have the same orientation.

Plugging in Your Sideways RAM

The UserRAM chip is even more fragile than an EPROM and should be treated with great care. Do not remove it from its protective foam until the last possible moment. If possible, hold it without touching the metal pins. Arrange the BBC micro so that the keyboard would be closest to you if the case was assembled. Hold the UserRAM so that the Red Wire is on the right hand edge; this is the right way round to fit into the BBC micro. Gently place the UserRAM in position on the chosen socket. DO NOT PUSH IT IN UNTIL YOU HAVE LOOKED CAREFULLY TO CHECK THAT ALL OF THE LEGS WILL PLUG IN CORRECTLY. Push the UserRAM home gently, making sure that pressure is applied evenly over the surface of the RAM. If you just push in the middle of the chip, you can bend the carrier board, and may damage the RAM. If you push on one edge, you may bend some of the pins. Be gentle and you will find it easy to push home.

Attaching the red lead

The red lead attached to pin 27 of the UserRAM must be attached to the a signal inside the BBC microcomputer which provides the timing information to enable writing to the UserRAM. (This signal is called "NWDS" on the circuit diagram. Normally, of course, ROMs must be written to and therefore this signal is not needed, which is why it is not available on the socket into which the UserRAM is plugged.)

On Boards less than Issue 10, the most convenient place to connect the red clip is to Pin 8 of IC 77. IC77 is the small chip (14 pins) 4 centimetres from the left hand edge of the board and 8 centimetres from the back edge of the board. It should have the legend 74LS00 included in the writing on top of the chip, although this may be part of a longer sequence such as SN74LS00N or DM74LS00N. Pin 8 is the pin on the corner of IC77 closest to the ROM sockets, ie on the right at the end nearest you as you look at the computer from the keyboard side (see figure 2 below).

On boards of issue 10 or later (such as the B+), the most convenient point to get NWDS from is Pin 6 of IC27. This chip can be found 10 cm from the left-hand edge of the board and 12 cms from the back. The correct chip should also have 74LS00 included in the legend on the top.

Dot and/or Half-moon

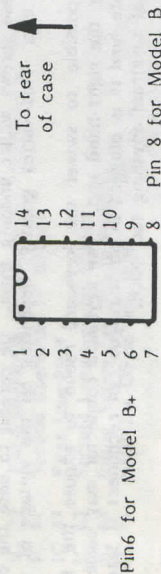


Fig. 2 Numbering of 14 pin chip.

View from top of board, as seen from keyboard.

Be careful when attaching the clip to the IC pin, as the reliability of the connection depends on the shape of the hooked part of the plunger, which is easily bent if you force it. The best method is to hold the clip with the plunger depressed and the open end of the hook uppermost. Place the hook next to the pin to which you are attaching it and gently rotate the whole clip, so that the top of the hook slips behind the IC pin. Now gently make sure that the clip is pressing against the side of the IC pin whilst releasing the plunger. Inspect the result closely to make sure that the metal part of the clip is not touching any other IC pin. With care this is not as difficult as it may sound.

Using More than One RAM

If you want to install more than one UserRAM, you will need to attach the write line to each of the RAM chips. There are several ways of doing this. It is usually possible to connect two clips to the same pin, but this is not very mechanically stable. It is better to use another IC with the same signal on it. Possible points to attach the red lead are:

- Model B : IC 71 pin 24 (5th pin from the top on the right hand side)
- IC 78 pin 10 (10th pin from the top on the left hand side)
- Model B+ : IC 84 pin 24 (5th pin from the top on the right hand side)
- IC 23 pin 2 (2nd pin from the top on the left hand side)

Reassembly

Putting your micro back together is the reverse of taking it apart. It should be safe to check the correct operation of the machine before screwing it together, provided the lid is correctly in place. Do not do this if there are people around (especially young children) who could be tempted to "investigate" whilst the machine is plugged into the mains, and do not forget to disconnect it from the mains before leaving it in this state, or before finishing the reassembly process. (Note that the machine will not work at all without the keyboard plugged in).

The keyboard on later machines has two narrow slots one in each side of the brown (or sometimes green) circuit board. These must be located over the corresponding lamps moulded into the lower half of the case. Be careful to poke the three red LED lamps through their correct holes in the case top.

If some of the keys round the edge of the keyboard stick down after you have reassembled the case, this may be because you have not positioned the keyboard correctly, or it may be because the lid is not fully home. Look round the edge of the case to check that the join is close fitting, and loosen of the case screws to re-adjust it if necessary, otherwise you will have to remove the lid again and slacken the keyboard fixing screws.

Testing

When you have reassembled your computer, insert the UserRAM utilities disc and type *RLIST. Your new UserRAM should be reported as Clear RAM.

Typing *RTEST will start a test routine which should report no errors. Let this run for a few minutes to check that the RAM works reliably.

Now type *DEMO to use your UserRAM in earnest for the first time.

If the test fails . .

Open your micro and inspect your installation carefully. In particular, check that the red wire has been connected to the right place, and that you have not bent any of the pins on the UserRAM. Also check the condition of the socket, in case any of the contacts are distorted or missing.

Pitfalls for the unwary!

In normal use there should be very few problems associated with your UserRAM. Most of the problems will occur when you attempt to use a copy of a ROM for the first time, when you may be thwarted by protection schemes. The kinds of effects which can occur are described later in this section.

The only other problems you could encounter are related to powering up your BBC Micro. These are not serious and are easy to avoid if you understand what is happening.

Externally powered peripherals

If you have peripherals attached to your BBC micro which have their own mains plug, it is possible for the BBC micro to be slightly powered from the peripheral, through the data cable. We have found disc drives with their own power supplies to be particularly effective at this. The result is that there is sufficient power available to do a passable job of retaining data, even when the BBC micro is switched off! Unfortunately, the data retention is not perfect, so that what can happen is that the OS believes that the ROM is valid, but the code is in fact slightly corrupt. The net result is usually a BBC micro which crashes when you turn it off and on again.

There are several solutions. Firstly, you can *KILL all your RAMs before turning off your machine. Secondly, you can turn off your peripherals too whenever you turn off your BBC Micro. This is a good thing to do anyway, especially if the BBC micro is going to be off for any length of time, because otherwise current will be flowing into the BBC micro through the outputs of some of the BBC Micro's chips.

Rapid cycling of the power switch

If you switch your BBC micro off and then switch it back on again very quickly, it is possible to get an effect similar to the external power problem. The BBC micro power supply will take time to turn off completely, and the RAM data may only change slightly in the first few seconds. If this happens it can easily be cured by turning the BBC micro off for ten seconds or so. Experimentation will soon show you the kind of times needed (but beware that some ROMs may be more persistent than others, due to the particular data patterns within them).

Protected ROMs

Many software publishers feel that allowing their software to be run in sideways RAM will increase the number of illegal copies in use. Unfortunately there is some evidence to show that they may well be right. We the public have only ourselves to blame that publishers are forced to waste valuable code-space on such unproductive activities.

If a ROM is protected, the publisher may be able to offer you an un-protected version, especially if you need a licence to run the software in a number of machines as is often the case in schools and colleges for example. You should contact the ROM publisher directly for details of this type of licence.

The methods of protection used are many and varied, and we certainly have no intention of explaining how any of them work, or how to get round them. In some cases the protection is so integrally tied up in the code that it is almost impossible to crack, and would certainly be easier to rewrite the ROM from scratch. From the point of view of users, it is only important to know how to detect the action of protection, so that you don't spend too long blaming your system or the UserRAM. Many such schemes rely on altering the contents of the RAM before starting the main code. The following procedure will detect this case.

1. Check that your UserRAM is working by using *RTEST
2. *RLOAD the ROM under investigation
 - Some protected ROMs may crash at once, or the next time you do a star command. Give up at once in this case!
3. Verify that the UserRAM is working by checking the image with *RCHECK

4. Try using the ROM

- There are plenty of ways of protecting the ROM at this stage. Some will change the memory, some will not. Some may crash the machine, some will simply not operate as expected. Be warned that some may try to damage data in the disc drive or on your file server!

5. Check that the UserRAM is unchanged using *RCHECK again

6. If the UserRAM is still unchanged but the ROM is not functioning correctly, there may be some very cunning protection which is not detectable with this method. Talk to the publishers or suppliers if you are unsure. Remember to check that it works correctly in ROM before making a fool of yourself on the telephone - it could be that you have not fully understood how the ROM is supposed to work.

If you wish to use protected software, get in touch with the manufacturers and ask them for a version which allows you to run in sideways RAM. There may be a premium payable for such versions since they are easier to make illegal copies of, and you may be required to give an undertaking that you will not make such copies.

Please don't steal copies of ROMs. Software houses have to earn a living just like everyone else and the more money they can plough back into new products the better will be the software that they can produce.

Acorn User Support

Acorn User will be offering continuing support for the UserRAM. Richard Harris' three articles in the June, July and August 1985 issues explain how machine code may be written to the sideways specification. In addition he provided some insight into how to adapt machine code routines published in Acorn User for use with the UserRAM. Where possible Acorn User will be providing conversion notes for machine code routines. In addition these routines will be supplied on the monthly cassette and in the Bar code booklet. Thus you will be able to build up a large and comprehensive set of machine code utilities for use in the UserRAM. If possible it is hoped to supply a 16k ROM image compilation on tape from time to time.

For details of support from Acorn User and software houses, see the latest issue of Acorn User magazine, or write to:

Acorn User,
RAM Support,
68 Long Acre,
London WC2E 9JH.

The UserRAM Pack 1 Utility Programs - Detailed Specifications

copy40

Purpose

Special BASIC program for reading 40 track discs on an unmodified 80 track drive, allowing you to make an 80-track copy.

Syntax

CHAIN "COPY40"

Example Output

CHAIN "COPY40"

```
40- to 80-track Disc Transfer V0.14
=====
```

```
**** WARNING **** This program destroys
the previous contents of the destination
disc!
```

Drive to copy from ? 0

Drive to copy to ? 1

Copying from Drive 0 to Drive 1

Copy Completed

>

Detailed Explanation of Operation

COPY40 copies all files on a 40-track disc onto an 80-track disc. The destination disc must be formatted before use. The copies are made on a sector-by-sector basis, ie in the same way as *BACKUP <S><D>. Existing data on the destination drive are deleted.

COPY40 uses the user memory as buffer space, and will therefore select screen MODE 7 if necessary.

Use on Model B+

Because the Model B+ DFS has the ability to read 40 track discs on an 80-track drive, COPY40 will never be needed on the B+. (It will not work as it assumes the old design of disc interface hardware).

*RLIST

Purpose

This program is a Sideways ROM catalogue utility. It indicates the current status of each ROM:

Off Entry in ROM table is zero
On Entry in ROM table is non-zero

Valid ROM/RAM contains the string (c) at the correct offset
Clear ROM/RAM does not have the string (c)

*RLIST also indicates the ROM type: S indicates a Service ROM, L indicates a Language ROM.

The Title column displays the title provided in the ROM header. Your current language is flagged with a star (*). The pointer to the ROM's private workspace is also displayed. The state of ROMs which allow frugalling can be deduced from this (see Appendix 2).

ROM slots which contain neither RAM, nor a ROM with a copyright string "(c)" will be indicated as "Empty". The program tests whether each ROM slot contains RAM by attempting to alter the first byte of the copyright string. The original values will be restored after testing. RAMs with no "(c)" will be reported as "Clear RAM".

If ROMs ghost into more than one ROM slot this will be reported as =nn=, with nn being the number of the highest ROM with similar contents. The program uses the same algorithm as OS 1.2 to determine whether two ROMs are actually the same, which compares the first &400 or so bytes.

Syntax

*RLIST

Example output

```
*rlist
No  Status/Type  Wkpg Title
15* (On ) Valid ROM L  BASIC
14  (On ) Valid ROM S &19  DFS,NET
13  (On ) Valid ROM S      TESTROM2
12  (Off) Clear RAM
11  (Off)          =15
10  (Off)          =14
09  (Off)          =13
08  (Off)          =12
07  (Off)          =15
06  (Off)          =14
05  (Off)          =13
04  (Off)          =12
03  (Off)          =15
02  (Off)          =14
01  (Off)          =13
00  (Off)          =12
```

Workspace used

RLIST: Two pages from &FFFF0900 (RS423, cassette, sound and speech workspaces)
User program memory should be unaffected.

*DEMO and R.DEMO

Purpose

*demo is the loader program for a sideways ROM called R.DEMO. To start the program, type *DEMO.

This ROM image contains a brief description of each of the UserRAM utilities, which can be accessed from a menu. When started, a main menu will be displayed, from which information about any of the utilities may be chosen. There is also an option to display a release note, which will be updated whenever changes are made to the other programs on the disc, or if any errors are found in this manual. *DEMO therefore gives the most up-to-date information available about your versions of the UserRAM utilities.

The R.DEMO ROM image also contains the code to support sideways ROM filing system files. Files saved in this ROM can be catalogued or accessed by typing *ROM, followed by *CAT, *LOAD <filename>, *RUN <filename> or LOAD" <filename>" as required. To return to your original filing system, type *DISC, *TAPE or *NET as appropriate.

Running the DEMO will also help to verify that your RAM is working correctly.

Syntax

*DEMO

Workspace used

*DEMO: Two pages from &FFFF0900 (RS423, cassette, sound and speech workspaces). R.DEMO will be loaded into any available RAM.
User program memory should be unaffected.

Common Errors

See documentation on files generated by MakeLD

*KILL [<ROM number>]

Purpose

*KILL removes the specified ROM number from the ROM table, and overwrites the copyright string (if possible) to prevent its reappearance. If no ROM number is given, *KILL removes all RAM-based sideways ROMs except your current language.

Syntax

*KILL [<ROM number>]
or *KILL

Examples

*KILL 14
*KILL

Workspace used

KILL: Two pages from &FFFF0900 (RS423, cassette, sound and speech workspaces)
User program memory should be unaffected.

Common Errors

nn is already dead

The specified ROM is not in the ROM map.

Can't kill nn

The specified ROM cannot be killed.

nn is your current language

You are trying to kill your current language in a tubeless computer. Select another language (eg *BASIC) and try again.

Bad number

The parameter given is not a valid decimal number in the range 0 to 15.

Detailed Explanation of Operation

*KILL can take a single parameter, which is the number of the ROM to disable. If no parameter is given, the program will attempt to overwrite the copyright message in all the ROMs (except your current language if no second processor is present), and will remove from the ROM table all the sideways RAMs which it succeeds in overwriting. The program can also be used to temporarily disable individual ROMs, using the *KILL <number> syntax, but in this case the ROM will re-appear when <Break> is pressed. *KILL-ing your current language is not allowed, unless you have a second processor attached.

*KILL also checks the table of ROM indirections, to make sure that none point to a disappearing ROM. If the filing system control vector (FSCVEC) points to any of the ROMs being disabled, that ROM is called upon to relinquish control cleanly. Any indirections which are still pointing to a KILL-ed ROM are reset to the OS default values. If one of the KILL-ed ROMs had control of the NMI, this will be claimed from it, but you will have to press CTRL-BREAK to sort out private workspace and IRQ usage.

***RLOAD**

Purpose

General load routine to copy any file into sideways RAM.

Syntax

*RLOAD <filename> [<ROM number>]
or *RLOAD

RLOAD needs to be able to load RLOAD2 in order to work.

Examples

*RLOAD
*RLOAD R.TOOLKIT 13
*RLOAD \$.LIBRARY.GRAPHICS
*load MyRom 14

Workspace Used

RLOAD: One page from &FFFFFF0C00 (user defined character definitions)
RLOAD2: Two pages from &FFFFFF0900 (RS423, cassette, sound and speech workspaces)
Note that any user defined characters will be corrupted by these programs, but user program memory should be unaffected.

Common Errors

RAM unavailable
Your machine has no RAM fitted, or all RAMs are in use. Try *KILL to free a RAM.

Bad ROM number
The second parameter is not a valid decimal number in the range 0 to 15.

RAM write error
Results if an error occurs when writing to the selected RAM. This can only be due to a hardware fault of some kind.

Not Found
Insufficient Access
etc

These error may be generated if RLOAD, RLOAD2 or the file you are trying to load cannot be found, or is not loadable for some reason.

Strange operation of loaded ROMs
ROMs which need to claim memory will not work correctly unless Control-BREAK is pressed after they have been loaded. In some cases it may be necessary to type *fx 151,78,127 before pressing the BREAK key, which forces a power-up Break on the IO processor.

Detailed Explanation of Operation

*RLOAD may take one or two parameters, which must be separated by spaces. If no parameters are given, the program will prompt you first for the filename and then for the ROM number. You may type <Return> in response to the request for a ROM number, in which case the program will search for sideways RAM as described below.

The first parameter will be interpreted as the name of the file which will be *LOAD-ed into the main RAM and then copied into the appropriate ROM number. Filenames should not be longer than 80 characters. RLOAD does not assume that the ROM code is stored in any particular directory, so the full filename must be given (e.g. R.VIEW, not just VIEW).

If only one parameter is given, RLOAD will automatically search for a free sideways RAM. The RAM is considered "free" if (a) it does not contain a copyright message, or (b) it is a language ROM which is not currently selected (if a second processor is attached language ROMs are never considered active, as overwriting them will not crash the machine). When searching for a free RAM, service ROMs are always considered active, as they may have intercepted some of the OS vectors, in which case overwriting them could crash your machine. If you wish to overwrite a service ROM, you must *KILL it first. If there is no free RAM an error will be caused: "RAM unavailable".

If two parameters are given, RLOAD will try to interpret the second parameter as a decimal number. Numbers outside the range 0..15 will cause an error ("ROM number?"). If the given ROM number is not sideways RAM, this will also cause an error ("RAM unavailable"). RLOAD will never allow you to overwrite your current language unless you have a second processor active, as you could be loading a service ROM and that would leave you without a language to return to.

RLOAD operates as follows:

1. RLOAD loads in RLOAD2, which contains the second part of the program.
2. The RAM destination is found.
3. A 16Kbyte buffer area from the main RAM (from &FFFFFF3C00 upwards) is swapped into the sideways RAM area to preserve it.
4. An attempt is made to *LOAD the specified filename to &FFF3C00.
5. Whether or not the attempt succeeded, the memory in the sideways RAM and the main RAM buffer area are exchanged again. If the *LOAD failed, this will leave the sideways RAM and the main RAM unchanged. On the other hand, if the *LOAD succeeded, the main RAM will have been restored, and the new ROM copied into the sideways RAM.
6. If the *LOAD failed, the error will be reported AFTER the second swap has taken place. This is necessary in case a mode other than MODE 7 was in use, in which case the error message would otherwise overwrite the buffer copy of the sideways RAM.
7. If the *LOAD did not generate an error, the arrival of the new ROM is announced to the MOS.

MakeLd (and *<romname> [<ROM number>])

Purpose

MakeLd is a BASIC program which generates bespoke star command programs for loading a particular sideways ROM program.

Syntax

CHAIN"MAKELD"

Output Generated

A star command file with similar function to *RLOAD, but with a fixed filename, e.g.:

*IO
*LOGO

(See the description below for the exact action of the file which is generated)

How to use MakeLd

Before using MakeLd, use *RSAVE to save a copy of the ROM you wish to use. Usually the ROM image should be saved as R.<romname>, (or \$.library.ROM.<romname>) if you have a level 2 Econet system). The load and execute addresses of the ROM image file are not important.

When you chain "MakeLd", it will ask you for details as follows:

Selection name for ROM?

Enter the name under which you wish the generated file to be saved. On disc this will normally be the same word that is used to start the ROM when it is installed, for example "LOGO" or "IO". On level 2 Networks you may find it convenient to enter the full pathname of the command as a library file, for example \$.LIBRARY.LOGO or \$.LIBRARY.IO

Name of ROM image file?

You are free to enter any valid filename here for the name under which you have saved the ROM image. The entered text will be incorporated into the generated program as the filename to be LOAD-ed. Net users should use the full name, ie "<discname>\$.LIBRARY.ROM.<romname>".

Is it a service ROM (S)
or a language ROM (L)

This defines whether the ROM will be activated (using *ix 142n) after it has been loaded. It also affects whether the generated star command is allowed to overwrite the current language RAM. (If your current language is in the RAM, overwriting it with a service ROM would otherwise leave you with nowhere to go. On the other hand, if you are loading a new language you don't care about this as you are about to start the new ROM instead anyway.)

What action should be taken if the

Tube is on?

- 1) Continue as normal
- 2) Load alternative code
- 3) Inhibit use

Choice (1-3)?

This question only applies to language ROMs.

Normal action is to load into sideways RAM and start the language using Osbyte &8E, whether or not a second processor is present.

The Load alternative code option allows a different file to be *RUN if a second processor is present (for example HIBASIC). In this case, when a second processor is present the sideways RAM will not be loaded. (The two filenames can be the same, in which case the only difference between 1 and 2 will be whether the sideways RAM is changed). The program will prompt "Enter name of alternative code?". Pressing <Return> will default to *RUN-ning the same filename.

Inhibit use on second processor is self explanatory. It should be used for any language ROMs which will not work over the tube.

After all these questions have been answered, the program will assemble a Star command with the appropriate characteristics. This will then be saved with the given name.

Common Errors

File too Big - Save anyway (y/n)?

This error will occur if very long (usually network) filenames are given for the name of the ROM image files. If the program generated in such cases is used, the user's soft key definitions will be corrupted.

Detailed Action of a Command Program file Assembled by MakeLd

*<filename> takes no parameters. If parameters are given, they will be ignored.

The command program operates in exactly the same way as *RLOAD <ROM image name>, except:

1. The command does not need to load a second program file, only the ROM image will be loaded by the command program.
2. If the command has been assembled for a language, it will be allowed to overwrite your current language. It will also start the new language, using *ix142n.
3. If no suitable RAM is found, the error message "RAM unavailable" will be accompanied by the suggestion "Try *KILL"

Workspace Used

The command program will load at &FFFF0900. Its length will depend on the filenames which were included when it was made.

Common Errors

As for RLOAD, except that the message

RAM unavailable

(Try *KILL)

is produced when RAM is unavailable. This is so that novice users do not have to refer to documentation when they have no available RAM.

***RSAVE**

Purpose

Star command to save a copy of the specified ROM number to the currently selected filing system (including Disc, Network File Server or Cassette), using the filename given.

Syntax

*RSAVE <filename> <ROM number>
or *RSAVE

Examples

*RSAVE
*RSAVE R.10 12
*rsave \$.library.r.basic 15

Workspace Used

RSAVE: Loads from &FFFFF2C00 upwards, with a buffer at &FFFFF3C00.
*RSAVE selects MODE 7 and destroys the contents of main RAM from &FFFFF2C00 upwards.

Common Errors

Bad ROM number
The second parameter is not a valid decimal integer in the range 0 to 15.

Detailed Explanation of Operation

*RSAVE usually takes two parameters, which must be separated by spaces. If no parameters are given, the program will prompt you first for the filename and then for the Rom Number.

The first parameter will be used as the name of the file to be saved. The FULL filename should be given (e.g. R.10 or \$.LIBRARY.ROM.LOGO, not just 10 or LOGO) as RSAVE does not assume that the ROM code should be saved into any particular directory.

If only one parameter is given, *RSAVE will prompt you for the ROM number to be saved.

RSAVE will try to interpret the second parameter as a decimal number. Numbers outside the range 0..15 will cause an error ("Bad ROM number"). If there is no "(c)" copyright message in the specified ROM slot the program will prompt with "No ROM found, save anyway? (Y/N) ". Pressing any key other than "Y" or "y" will abort the program.

The ROM is copied into a 16Kbyte buffer area of main RAM from &FFFFF3C00 upwards. An attempt is made to *SAVE the buffer area with the specified filename. The load and execute addresses are always set to &00008000, to enable the generated ROM image to be *RUN on second processors.

***RCHECK**

Purpose

*RCHECK checks the contents of the given ROM number against the specified filename and reports any differences that are found. This is useful to detect self-corruption by protected ROMs.

Syntax

*RCHECK <filename> <ROM number>
or *RCHECK

Examples

*RCHECK
*RCHECK R.10 12
*rcheck \$.library.r.basic 15

Workspace Used

RCHECK: Loads from &FFFFF2C00 upwards, with a buffer at &FFFFF3C00.
*RCHECK selects MODE 7 and destroys the contents of main RAM from &FFFFF2C00 upwards.

Common Errors

Bad ROM number
The second parameter is not a valid decimal integer in the range 0 to 15.
Insufficient Access, Not Found, etc.
Errors produced by the filing system.

Detailed Explanation of Operation

*RCHECK normally takes two parameters, which must be separated by spaces. If no parameters are given, the program will prompt you first for the filename and then for the Rom Number. If only one parameter is given, *RCHECK will prompt you for the ROM number to be checked.

The first parameter will be used as the name of the file to be checked against the ROM.

*RCHECK will try to interpret the second parameter as a decimal number. Numbers outside the range 0..15 will cause an error ("Bad ROM number").

*RCHECK operates as follows:

1. The specified file is *loaded into a 16Kbyte buffer area of main RAM from OSHWM upwards.
2. The buffer area is compared with the ROM contents, and any differences are reported.

*RTEST

Purpose

RTEST checks that the given ROM number works properly as RAM.

Syntax

*RTEST [<ROM number>]
or *RTEST

Examples

*RTEST 14
*RTEST

Workspace Used

The program loads at &FFFF2C00 and corrupts user memory from there upwards.

Common Errors

Bad ROM number

The parameter entered is not a valid decimal integer in the range 0 to 15.

Detailed Explanation of Operation

*RTEST normally takes a single parameter. If no parameters are given, the program will prompt you for the ROM number of the sideways RAM that you wish to test.

When first run, RTEST selects MODE 7 and prints a title banner. The program fills the whole of the sideways RAM with a test pattern, and then reads it back to check that it was correctly saved. After each successful pass the program will print a message.

Unless an error is detected, the program will repeat using a variety of different data patterns until <Escape> is pressed. Control will then return to the user. Any error will stop the program.

This program should be used at least once, when the RAM is first installed (see installation details), and at any time when incorrect operation of the RAM is suspected.

MakeRFS

Purpose

To make a ROM filing system (RFS) image file for the specified program or data files.

Syntax

CHAIN"MakeRFS"

Output Generated

A file which can then be *RLOADED, or blown into an EPROM

How to use MakeRFS

The Image File generated by MakeRFS will be assembled in RAM and then *SAVEEd, unless there is not enough space to fit the whole image file. In this case the files will be loaded and saved in blocks, which may be difficult to use with cassette-based systems, or with non-Acorn DFSs which do not support the block transfer protocol properly. The normal mode of operation will be to assemble the file in RAM, which should work properly with all filing systems.

When MakeRFS is run, it will first prompt you for details of the ROM image you want to generate. Star commands can be executed at any time, by typing "<commandline>", for example *disc or *TAPE.

Image File Name:

Enter the filename as you wish it to be saved on the disc.

Image File Length in Kbytes (range 1..16)

(Press <Return> for 16 K) :

Normal values are 4, 8 or 16, but any positive integer up to 16 may be entered. If you enter a value less than 0 or greater than 16 the program will ask you to re-enter a valid number.

The program will now assemble the appropriate RFS header, and enter a loop which prompts for new files to be included in the ROM image.

File Name to Load (<Return> to Exit):

Enter the name of the next file to be included in the ROM. Pressing <Return> will finish the loading process and save back the ROM image.

Name in ROM (if different):

Press <Return> unless you want the filename in the ROM to be different from the name of the file you have just loaded, in which case enter the new name.

The program attempts to read the length of the file. The program does not support splitting files between ROMs. If the file is too long, an error will be reported:
File too long.

If it is not possible to read the length of the file (for example on the Cassette Filing System) the program will ask you to enter the length of the file. This should have been previously found out by doing a *OPT 1 2 followed by a *CAT.

When you have entered all the filenames you wish to include in the ROM image, press RETURN in response to the "Type file name -" prompt. The program will then ask you for the name with which you wish to save this image file, and the image file will be SAVED with the appropriate length.

Appendix 1 :- Summary of the Software supplied with UserRAM

Pack 1

- Copy40 - Special Basic program for reading 40 track discs on an unmodified 80 track drive, allowing you to make an 80-track copy.
- *RLIST - Produces a list of currently installed sideways ROMs.
- *DEMO (With R.DEMO) - a sideways RAM demonstration.
- *KILL - makes the specified ROM invisible to the BBC micro.
- *RLOAD (With RLOAD2) - General purpose routine to copy any file into sideways RAM.
- MakeLd - Generates a bespoke star command program for loading a particular sideways ROM program
- *RSAVE - Saves a copy of the specified ROM to the current filing system.
- *RCHECK - Checks the contents of a ROM against a specified filename, and reports any difference that are found.
- *RTEST - Checks that the RAM works correctly
- MakeRES - Produces a ROM image in sideways ROM filing system format.

Pack 2

As pack 1 but with the following additions:

- B.AURAM - Loader Program for the Acorn User RAM utilities image (R.AURAM). These include an I1K printer buffer; safe *COPY, *BACKUP, *BUILD and *COMPACT commands (ie they do not destroy current memory contents); Disc formatting and Verifying routines; Dual disc catalogue allowing 61 files per disc; ROM handling routines; paged mode facility; Full *HELP and error handling.

Full details on using these utilities appeared in the June, July and August 1985 issues of Acorn User.

The other programs are included from The BBC Micro ROM Book by Bruce Smith:

- B.RDROM - Read a byte from a ROM (page 10)
- B.BRKCMD - The BRK interpreter (page 44)
- B.NELP - Implementing *HELP (page 90)
- B.HBTOOL - An extended *HELP (page 100)
- B.TESTINT - Testing the interpreter (page 102)
- B.BTOOLS - The complete interpreter (page 111)
- B.TITLE - Print title string on BREAK (page 124)
- B.PRIVATE - Claiming workspace (page 126)
- B.CBREAK - Cat disc on C-BREAK (page 127)
- B.OSWORD - Adding an OSWORD call (page 130)
- B.OWTEST - Testing the new OSWORD call (page 133)
- B.OSBYTE - Adding an OSBYTE call (page 134)
- B.OBTEST - Testing the new OSBYTE call (page 135)
- B.TREK - A language ROM (page 136)
- B.EPROG - An EPROM programmer (page 168)
- B.VEPROM - Verify an EPROM (page 171)
- B.CEPROM - Copy EPROM (page 174)

Appendix 2 :- The Frugalling Protocol

What is frugalling?

Each sideways ROM slot has a single byte in which it can store the page number of the start of its private workspace. Since it is very unlikely that anyone would want to use a BBC Micro with PAGE permanently above &4000, this number is invariably less than &40. This means that there are two "spare" bits in this byte, which can be used for other purposes. The two spare bits are value &80 and &40 respectively. ROMs which support frugalling use these bits to indicate whether they are "on" or "off", and they only claim workspace when they are on.

The workspace pointer is then used like this:

```

bit 7 bit 6 bit 5 bit 4 bit 3 bit 2 bit 1 bit 0
Flag A Flag B <----- Start-of-Workspace pointer ----->

```

In the DNFS for example, Flag A is used by the Net and Flag B is used by the DISC.

In all cases, 0=ON, 1=OFF

By writing the appropriate value to the ROM workspace table and then pressing Control-BREAK, the corresponding Filing system can be made to disappear. It will not come back (even at Control-BREAK) until either the ROM workspace pointer is changed again, or the BBC Micro is turned off.

The ROM workspace pointers are located from &FFFF0DF0 (for ROM number zero) to &FFFF0DFF (for ROM number 15). Remember that they will always be in the IO processor, and therefore cannot be accessed directly from BASIC if you have a second processor attached. In that case you will have to use the appropriate OSWORD call to read or write IO-processor memory.

Note that it is ESSENTIAL to press Control-BREAK, not just BREAK, even though BREAK may appear to work. If you don't, the subtle differences may come back to haunt you at any time.

How to Frugal ROMs under software control

There is often very little justification for frugalling ROMs, as with a little more thought and careful crunching of your program it is usually possible to do the same thing in less space (although there is obviously a limit to this). Frugalling almost always causes undesirable side effects (for example frugalling the Net means that inter-machine communication is disrupted).

NOTE: CALLING !-4 (ie JMP (&FFFC)) WILL NOT ALWAYS WORK. DON'T DO IT!

The frugalling rules state that you must always press Ctrl-BREAK after a ROM has been frugalled or de-frugalled. If you simply press BREAK, many of the filing systems will get confused, as they will not bother to re-initialise their newly allocated workspace area. Jumping to !&FFFC is even worse, as all the software assumes that the hardware RESET line has been activated, which should have reset all the peripheral chips to a known (inactive) state. It is quite possible for the BBC Micro simply to crash under these conditions.